This is a readme file for the Vol 13 article *An Open-Source Electroacoustic Measurement System* by Richard Mann and John Vanderkooy.
This readme file applies to the .m files for part 1: *Theory, Practicalities & Acoustic Examples* by John Vanderkooy

This file is a condensation of the details in Part 1 as they relate to three of the Octave/Matlab measurement programs. In many ways reading the last portion of Part 1 is preferred. We assume that you will have Octave or Matlab open with the relevant program in the editor, ready to run. A soundcard needs to be present, and you should set its sampling frequency the same as in the program. Each program contains first segments that set things like sampling frequency, the total number of time samples, output level, sound card calibration, etc. You will see lines with a % that mark them as comments. Actually, if a soundcard is not present, the computer will try to use the internal loudspeaker and microphone, and show you how bad they really are!

Logsweep1quasi.m, is intended to measure a quasi-anechoic loudspeaker transfer function in a normal room. Since there are a lot of reflections from objects in the room, the program allows the measured impulse response to be edited to remove the reflections, so that the response closely approximates that which would occur in a real anechoic chamber. It is presented with commented lines that will help the understanding of each operation. The program is fairly verbose in its plotting of intermediate data, which can be commented out later when you get the hang of things. This helps to visualize the various views of the data as the program wends its way through, (0) preliminaries and settings, (1) calculating the logsweep, (2) data-gathering, (3) determining record/play delay, (4) calculating a transfer function with room reflections, (5) getting the previous TF and comparing them, (6) calculating the associated impulse response, (7) editing it to remove reflections, and finally (8) obtaining the quasi-anechoic transfer function response. The first 5 or 6 steps are essentially the same in each of the logsweep programs. The user will be prompted to click on three time positions of the impulse response to complete the calculation. Matlab will show you a nice crosshair at each point as you move the mouse, Octave has not implemented that yet, and simply shows the mouse position. If you are used to Matlab, you will sometimes find that a particular peripheral function is often just ignored by Octave. All the important things work!

Some tips on running the programs are in order. If the length of the recording file and sampling frequency are not appropriate, a multiple message "INSUFFICIENT TIME CLEARANCE" will be sent to the command window. It means that the record-play delay exceeds the allowed time clearance, which is printed onscreen soon after the program starts. You must then either reduce the sampling

frequency, or increase the index defining the power-of-2 size of the files. By changing the constants in the first part of the programs, such as *sig_frac*, you can also prevent overload and modify the environment.

Quasi-anechoic measurements have a number of limitations. The loudspeaker is normally placed on a stool about halfway between ceiling and floor, with the microphone on the measuring axis as close as practical, and distances of 40 or 50 cm are not unreasonable. This reduces the amplitude of the reflections relative to the direct sound.

The program pauses to display a zoomed portion of the loudspeaker impulse response, and requires three clicks from the mouse. The plot may display some acausal wiggles that result from the method of computing the impulse response and/or the AA filter. These should be included in the impulse response, wrapped in periodic time. Three time values are selected on the plot: the first should be such as to be just before any acausal wiggles, the second should be *time zero* where the impulse response rises sharply, and the third must be chosen so that room reflections are removed beyond it. The program then recomputes the *reflection-free* transfer function and displays both the phase and magnitude quasi-anechoic responses.

If we truncate the impulse response after a time τ, the response will be smeared and not reliable below frequency $1/\tau$, typically around 200 Hz. However, we will indeed have removed the room reflections. In order to obtain a fairly good "anechoic" measurement at the lowest frequencies, we can use a nearfield technique, placing the microphone near the woofer dustcap. It is often possible to get meaningful nearfield results and "stitch" them onto those measured quasi-anechoically.

Logsweep1rt.m, calculates room reverberation time, RT, using a microphone placement that is usually much further from the loudspeaker, thus making the reverberation more prominent. Its data-gathering part is the same as Logsweep1quasi.m, and the program then calculates an acoustic room parameter, clarity, from the unfiltered impulse response. C50 (or C80) is the ratio of the early impulse response energy before 50 (or 80) milliseconds, compared to the remaining late energy, expressed in decibels. The impulse response with all the room reflections is used to compute the reverberation time. The program does a reverse integration of the square of the impulse response, which is the energy decay. If *h[n]* are the samples of the impulse response, the decay of the energy, *D[n]*, in the room is given by the wonderfully-compact Octave/Matlab statement,

$$D[n] = \text{flip}(\text{cumsum}(\text{flip}(h^2[n])));$$

As the program nears completion, the user is prompted to enter the centre frequency of the octave band that is to be analyzed (you might have to position the cursor in the command window for Octave/Matlab to recognize this), and then the decay of the reverberant energy is plotted. After clicking with the mouse on two points of the plot where the decay is fairly straight, the reverberation time is calculated. Although the decay of the energy is usually much less than 60dB, the reverberation time is always scaled to represent a full decay of 60dB.

Logsweep1hd.m, computes the linear transfer function of a system together with spectra of the second and third harmonic distortion. It is intended for loudspeakers, and typically the microphone should be placed very close to the speaker, to minimize the reverberation of the room, while capturing all the relevant loudspeaker distortion details. The logsweep has a clean sinusoidal waveform, and the distortion will be solely harmonic, being captured by the microphone together with the linear response. Since the logsweep covers each octave in the same period, the second harmonic will be visible in the recovered impulse response as a relatively clean pulse, *advanced* by the time it took for the sweep to cover a factor of two (1 octave) in frequency. The advance occurs because when the logsweep is at frequency *f*, the second harmonic is already at frequency 2*f,* so its impulse will be recovered early. Similarly the third and higher harmonics will also be separated out early at the respective times that the logsweep took to cover that frequency interval.

The harmonic impulse responses must be extricated from the total impulse. Each one must include the acausal wiggles that are caused by the AA filter and/or measurement artifacts, and the clean part that follows, without including bits from the other harmonics. The higher harmonic impulses are shorter and therefore more difficult to extract due to the compressed time scale as the order increases. The program is tricky, but careful reading of the lines and comments will guide you through.

I urge you to modify the programs to personalize the plots, change the parameters, and add new twists of your own. The **pwroctsmooth.m** program does a fractional-octave smoothing using a subtle algorithm. There are other smoothing programs as well that may be presented later.